Gerarchia di Chomsky

→ $L = \{a^n b^n \mid n \geq 1\}$ può essere generato da grammatiche di tipo 0 perchè ammette ε-produzioni sia da grammatiche di tipo 2 (non contestuali) non tipo 3

$L_0 \in$ tipo 0 ma potrebbe non appartenere a tipo 1|2|3

Osservazione:

Se analizziamo grammatiche di tipo 2/3 è consentita la presenza di una (più) ε-produzione, mentre nel caso di tipo 1 è ammessa una ε-produzione solo sull'assioma S, a condizione che questo non compaia mai nelle parte destra di una produzione

regolare: "possiamo procedere senza fermarci"

Grammatica **Tipo 3?**

$G: S \rightarrow aS \mid bA$

$\quad A \rightarrow bA \mid \varepsilon$

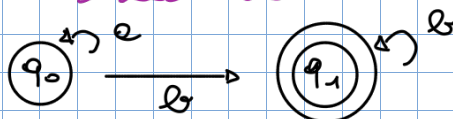
Togliamo la ε-produzione per fare esercizio

$G: S \rightarrow aS \mid bA$

$\quad A \rightarrow bA$

→ relativa espressione regolare

$a^* b b^*$ oppure $a^* b^+$

ASFD relativo

CAPITOLO 4: linguaggi non contestuali

Inizialmente queste sono state introdotte come strumento per definire le strutture del linguaggio naturale.

Possiamo caratterizzare le basi di un linguaggio con soggetto e predicato e quindi comporre una frase

$F \to SP$ frase $\to$ soggetto + predicato

$P \to V \mid VC$ predicato $\to$ verbo | verbo + complemento oggetto

Queste strutture sono sufficientemente semplici e quindi in alcuni contesti inadeguate come strumento per il linguaggi modulari.

I linguaggi generati non contestuali rappresentano un sottoinsieme proprio dei linguaggi

I linguaggi di programmazione rappresentano un sottoinsieme proprio dei linguaggi non contestuali e li caratterizziamo come alberi.

Alberi di derivazione o alberi sintattici

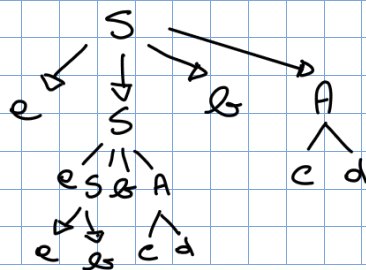
G: $S \to aSbA \mid ab$

$A \to cAd \mid cd$

$\to aSbA \to aabbA \to aabbcd \in \mathcal{L}(G)$

La grammatica introduce il linguaggio $\mathcal{L}(G)$ aaabbcdbcd $\in \mathcal{L}(G)$

un'esempio di albero di derivazione di una stringa $\in \mathcal{L}(G_1)$

```
            S
   /     /    \     \
  a     S      b     A
      / | | \       / \
     a  S b  A     c   d
       / \   / \
      a   b c   d
```

Queste grammatiche usano una memoria gestita a pila e nel caso di linguaggi di programmazione queste corrispondono ad un'analisi sintattica "parsing" relativa al programma di "tradurre" la stringa in un'altra stringa più elementare

capibile dal PC

CAPITOLO 5:

Macchine di turing e calcolabilità secondo turing

Le mdt sono il modello di calcolo di riferimento fondamentale sia nell'ambito della teoria di calcolabilità sia in quella della teoria della complessità computazionale

L'obbiettivo è quello di formalizzare il concetto di calcolo allo scopo di stabilire l'esistenza di metodi algoritmici per il riconoscimento dei teoremi nell'ambito dell'aritmetica

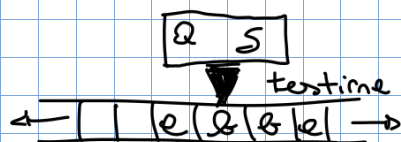
- Elevata semplicità strutturale
- Potere computazionale

La macchina di turing viene definita come un dispositivo operante su stringhe contenute su una memoria esterna (nastro) mediante passi elementari definite da funz. di transizione

Macchine di Turing a nastro singolo

Nella MT vi è un nastro potenzialmente infinito diviso in celle contenenti ciascuna un simbolo appartenente ad un dato alfabeto Γ (↳ gamma) ampliato con il carattere speciale $\bar{b}$ (blank) che rappresenta la situazione di celle non contenente caratteri

La MT lavora su un nastro, o vi è una testina che scorre su di esso in entrambe le direzioni.

Si può leggere o scrivere caratteri dell'alfabeto Γ oppure $\bar{b}$

Ad ogni istante la MT si trova in uno stato appartenente ad un insieme finito Q e il meccanismo che fa evolvere la computazione della macchina è la funzione di transizione

δ parte da uno stato e da un carattere → stato

Definizione MT

Una macchina di Turing deterministica (MTD) è una sestupla $M = \langle \Gamma, \bar{b}, Q, q_0, F, \delta \rangle$

Γ = alfabeto

$\bar{b}$ = blank $\notin \Gamma$ ("b sopra-segnato")

Q = insieme degli stati

q_0 = stato iniziale

F = stati finali $F \subseteq Q$

δ = funzione di transizione definita così:

$$\delta : (Q - F) \times (\Gamma \cup \{\bar{b}\}) \mapsto Q \cup (\Gamma \cup \{\bar{b}\}) \times \{d, s, i\}$$

(Q − F): STATO NON FINALE; (Γ ∪ {b̄}): SIMBOLO; Q: stato; (Γ ∪ {b̄}): simbolo; {d, s, i}: spostamento

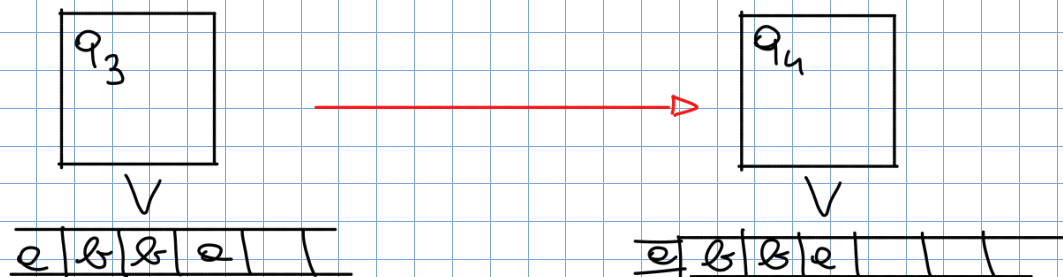
$\{d, s, i\}$ indicano lo spostamento a dx, sx o assenza di spostamento della testina

$\bar{\Gamma} = \Gamma \cup \{\bar{b}\}$ per convenzione

e soprattutto $\Sigma \subseteq \Gamma$

Esempio:

$$\delta : (Q - F) \times (\Gamma \cup \{\bar{b}\}) \to Q \times (\Gamma \cup \{\bar{b}\}) \times \{d, s, i\}$$

$(q_3, b) \mapsto (q_4, a, d)$

$\delta(q_3, b) = (q_4, a, d)$

Le macchine usate per accettare stringhe vengono dette riconoscitori, mentre quelle usate per calcolare funzioni vengono dette di tipo trasduttore

Funzione caratteristica del linguaggio che è definita da Σ^* a $\{0,1\}$ ed è ovviamente una funzione che assume valore 1 per ogni stringa del linguaggio o valore 0 altrimenti

Forma normale di Greibach

"$A \to a\alpha$" $\quad A \in V_N \quad a \in V_T$

$\alpha \in V_N^*$